Listing of Claims

Claim 1 (Currently Amended): A method of implementing ~~an~~ atomic transactions in a system ~~using a program logic~~, said method comprising:

requesting in ~~said~~ a program logic a transaction identifier for ~~said~~ an atomic transaction, wherein said program logic is contained in a user program designed by a programmer;

generating said transaction identifier in a transaction manager in response to said requesting;

specifying in said program logic a plurality of combinations for execution in a sequential order, wherein each of said plurality of combinations contains said transaction identifier, a task procedure, and a rollback procedure, wherein said task procedure implements a part of said atomic transaction and said rollback procedure is designed to rollback said task procedure;

executing said task procedures in said sequential order;

keeping track of said rollback procedures in said transaction manager; and

executing said rollback procedures in a reverse order of said sequential order if said atomic transaction is to be aborted, wherein said rollback procedures are identified according to said keeping.

Claim 2 (Original): The method of claim 1, wherein said transaction identifier is unique to each of the atomic transactions.

Claim 3 (Previously Presented): The method of claim 1, wherein said keeping comprises storing data representing said rollback procedures in a stack.

Claim 4 (Original): The method of claim 3, wherein said stack is stored in a memory.

Claim 5 (Original): The method of claim 1, further comprising examining a status returned by execution of one of said task procedures and performing said aborting if said status indicates an error.

Claim 6 (Original): The method of claim 1, wherein said aborting is performed asynchronously.

Claims 7 (Currently Amended): A computer readable medium carrying one or more sequences of instructions representing a program logic for execution on a system, said program logic implementing an atomic transaction, wherein execution of said one or more sequences of instructions by one or more processors contained in said system causes said one or more processors to perform the actions of:

requesting an identifier for said atomic transaction;

setting a variable to equal said identifier;

specifying a plurality of combinations for execution, wherein each of said plurality of combinations contains said transaction identifier, a task procedure, and a rollback procedure, wherein said task procedure implements a part of said atomic transaction and said rollback procedure is designed to rollback said task procedure; and

aborting said atomic transaction by specifying said identifier associated with an abort procedure to cause said rollback procedures to be executed,

wherein said program logic, including said plurality of combinations, are contained in a user program designed by a programmer.

Claim 8 (Original): The computer readable medium of claim 7, wherein said specifying comprises including each of said plurality of combinations in a single procedure call.

Claim 9 (Original): The computer readable medium of claim 7, further comprising examining a status returned by execution of one of said task procedures and performing said aborting if said status indicates an error.

Claim 10 (Currently Amended): A computer readable medium carrying one or more sequences of instructions for supporting implementation of an atomic transaction in a system, wherein execution of said one or more sequences of instructions by one or more processors contained in said system causes said one or more processors to perform the actions of:

generating an identifier for said atomic transaction;

receiving a plurality of combinations for execution, wherein each of said plurality of combinations contains said transaction identifier, a task procedure, and a rollback procedure, wherein said task procedure implements a part of said atomic transaction and said rollback procedure is designed to rollback said task procedure;

executing said task procedures; and

executing said rollback procedures in response to receiving an abort request,

wherein each of said plurality of combinations is received from a corresponding user program and wherein the roll back procedure received from a first user program is different from the roll back procedure received from a second user program.

Claim 11 (Previously Presented): The computer readable medium of claim 10, wherein said task procedures are executed in an execution order and corresponding rollback procedures are executed in a reverse order of said execution order.

Claim 12 (Previously Presented): The computer readable medium of claim 11, further comprising storing data indicating that said rollback procedures are to be executed in said reverse order to abort said atomic transaction.

Claim 13 (Previously Presented): The computer readable medium of claim 12, wherein said transaction identifier is generated to be unique for each atomic transaction.

Claim 14 (Original): The computer readable medium of claim 12, wherein said data is represented in the form of a stack.

Claim 15 (Original): The computer readable medium of claim 14, wherein said stack is stored in a memory.

Claim 16 (Currently Amended): A computer system comprising:

a memory storing a plurality of instructions; and

a processing unit coupled to said memory and executing said plurality of instructions to support implementation of ~~an~~ atomic transactions in a programming environment, said processing unit being operable to:

request in a program logic a transaction identifier for ~~said~~ an atomic transaction;

generate said transaction identifier in a transaction manager in response to said requesting;

specify in said program logic a plurality of combinations for execution in a sequential order, wherein each of said plurality of combinations contains said transaction identifier, a task procedure, and a rollback procedure, wherein said task procedure implements a part of said atomic transaction and said rollback procedure is designed to rollback said task procedure;

execute said task procedures in said sequential order;

keep track of said rollback procedures in said transaction manager; and

execute said rollback procedures in a reverse order of said sequential order if said atomic transaction is to be aborted, wherein said rollback procedures are identified according to said keeping,

wherein said program logic is contained in a user program designed by a programmer.

Claim 17 (Original): The computer system of claim 16, wherein said transaction identifier is unique to each of the atomic transactions.

Claim 18 (Previously Presented): The computer system of claim 16, wherein said processing unit is operable to store data representing said rollback procedures in a stack to perform said keep.

Claim 19 (Original): The computer system of claim 18, wherein said stack is stored in a memory.

Claim 20 (Original): The computer system of claim 16, wherein said processing unit is further operable to examine a status returned by execution of one of said task procedures and to perform said aborting if said status indicates an error.

Claim 21 (Previously Presented): The computer system of claim 16, wherein said processing unit is operable to execute said rollback procedures asynchronously.

Claim 22 (New): The method of claim 1, wherein different user programs contain different rollback procedures such that each programmer can specify custom rollback procedures in the respective user program for execution in said system.

Claim 23 (New): The method of claim 7, wherein different user programs contain different rollback procedures such that each programmer can specify custom rollback procedures in the respective user program for execution in said system.

Claim 24 (New): The computer system of claim 16, wherein different user programs contain different rollback procedures such that each programmer can specify custom rollback procedures in the respective user program for execution in said system.